2 Sep 2005 - DSD Department Mtng - LBNL

The Performance Evaluation Research Center (PERC)

Part 1: PERC Goals

- Outlines the participants and goals of the PERC project
- Content from David Bailey (dhbailey@lbl.gov)
- David's original talk (powerpoint, pdf)

PERC Project

- An "Integrated Software Infrastructure Center" (ISIC) sponsored under DoE's SciDAC program
- Approximately $2.4 million per year for 4 years.
- Four DoE laboratories
- Four universities
- Mission:
  - Develop a *science of performance*, and engineer tools for performance analysis and optimization. Focus on large, grand-challenge calculations, such as SciDAC application projects.

PERC Participants

- Laboratories:
  - LBNL (Bailey, Strohmaier)
  - LLNL (Quinlan, de Supinski, Vetter)
  - ORNL (Worley, Dunigan)
  - ANL (Hovland, Norris)
- Universities:
  - Univ of Tennessee (Dongarra)
  - Univ of Illinois (Reed)
  - Univ of Maryland (Hollingsworth)
  - UCSD Supercomputer Center (Snavely)

Feedback to Vendors

- HPC relies heavily on commercial vendors for high performance computer systems

- Researchers are invited by vendors to provide guidance on the design of current and future systems.

BUT..

- At present they can provide only vague information and little if any quantitative data or rigorous analysis.
- As a result, they have little voice in future designs.

SciDAC Applications

PERC is really only valuable if it helps SciDAC applications.

It's the Memory, Stupid

- For the foreseeable future, time to solution will be dominated by a code's ability to effectively utilize the memory hierarchy, including local and distributed memory.

Research Questions

- How can we best measure the memory hierarchy behavior of a particular code on a particular system?
- Can we construct accurate models of performance, based on data that is easily obtained?
- Can we accurately project the performance of a future version of a code on a future system?
- If we determine that a given code is running suboptimally, can we facilitate the necessary changes to improve performance?

Work Areas

- Benchmarks
- Performance monitoring tools
- Performance modeling and analysis
- Software tools to automatically or semi-automatically optimize user codes

PERC Summary

- Achieving optimal performance on HPC systems has compelling economic and scientific rationales
- Performance is poorly understood and in depth-studies do not exist except in a handful of cases
- PERC is pursuing "performance science" and "performance engineering", including improved benchmarks, monitoring tools, modeling techniques, and optimizers.

PERC future

- PERC project is up for renewal
- The renewal proposal will contain a superset of the original participants.
  - "nobody left to submit a competing proposal"
- Bob Lucas (ISI) has taken over managing the project
- Proposal is yet to be written, but it's clear that the focus will shift from tool development and

deployment to automatic performance tuning and analysis.

PERC future participants

Meeting next week in Oakland to plan the proposal, only 1-2 people per institution still leaves an impressive list:

- David H Bailey (LBNL), Bronis de Supinski (LLNL), Jack Dongarra (UTenn), Robert Fowler (Rice), Jeff Hollingsworth (Univ Maryland), Paul Hovland (ANL), Robert Lucas (ISI), Shirley Moore (Univ Tenn), Boyanna Norris (ANL), Dan Quinlan (LLNL), Dan Reed (Rice Univ), Sameer Shende (Univ Oregon), Allan Snavely (SDSC), Erich Strohmaier (LBNL), Jeff Vetter (ORNL), Pat Worley (ORNL), Kathy Yelick (UCB), Ying Zhang (UNC)

Part 2: LBNL work

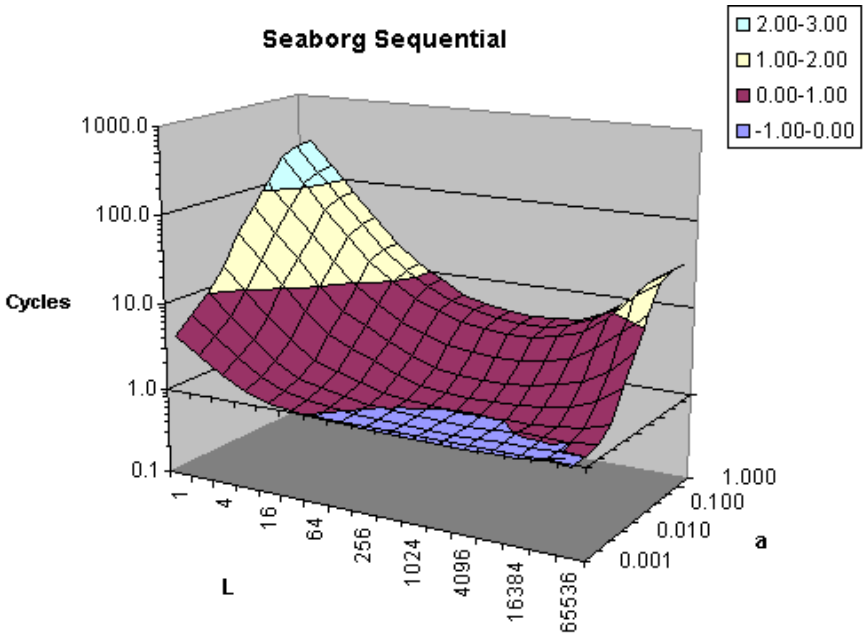
- Participants
  - David Bailey
  - Kathy Yelick
  - Erich Strohmaier
  - Honzhang Shan ("shan")

Erich Strohmaier/Hongzhang Shan’s work

- Tool called APEX-MAP that benchmarks memory performance.
- Considers both parallel and sequential performance for varying locality
  - *Spatial locality*: Distance in memory between subsequent accesses in time. If this distance is fixed, it’s called a "stride".
  - *Temporal locality*: Distance between access of the same address in time.

APEX-MAP examples (1)

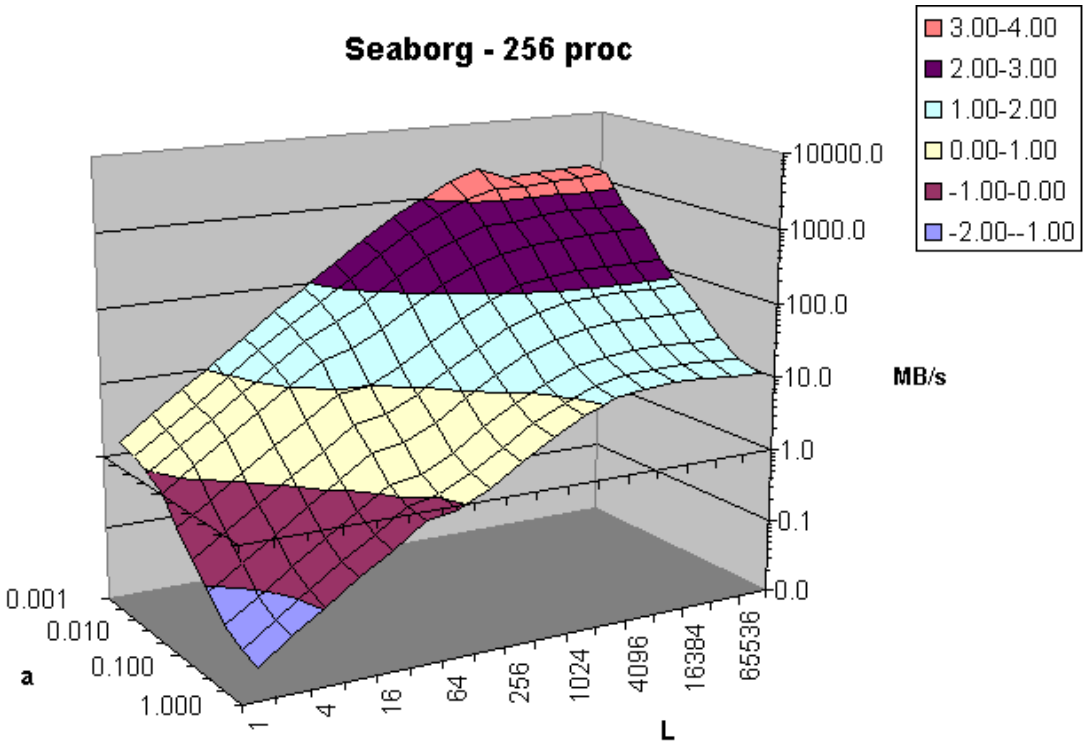
Sequential performance on Seaborg



- *L* stands for vector-length and represents spatial locality
- *a*represents temporal locality
  - `a=1` is a random walk
  - `a << 1` is high temporal locality.

APEX-MAP examples (2)

Parallel (MPI) performance on Seaborg



Kathy Yelick's work

Kathy is the group lead of FTG, but also a professor at UCB. Therefore, her projects involve both campus and NERSC people.

- UPC: C language extensions for HPC
- Titanium (with Phil Colella): HPC Parallel dialect of Java
- BeBOP (with James Demmel): Automatic tuning of linear algebra algorithms, particularly "sparse kernels" (SpMV) and some software called "OSKI".

Note: PERC doesn't directly fund the above research, yet. Kathy is officially just managing the LBNL participants.

David Bailey's work

- David is like a consultant to PERC. His main work, as we all know, is "experimental mathematics".

What have I been up to?

My work in PERC has mostly been on analyzing performance data from real applications, on the NERSC machines. I have worked with two applications:

- Omega3P
- BeamBeam3D

And used (or tried to use) these tools:

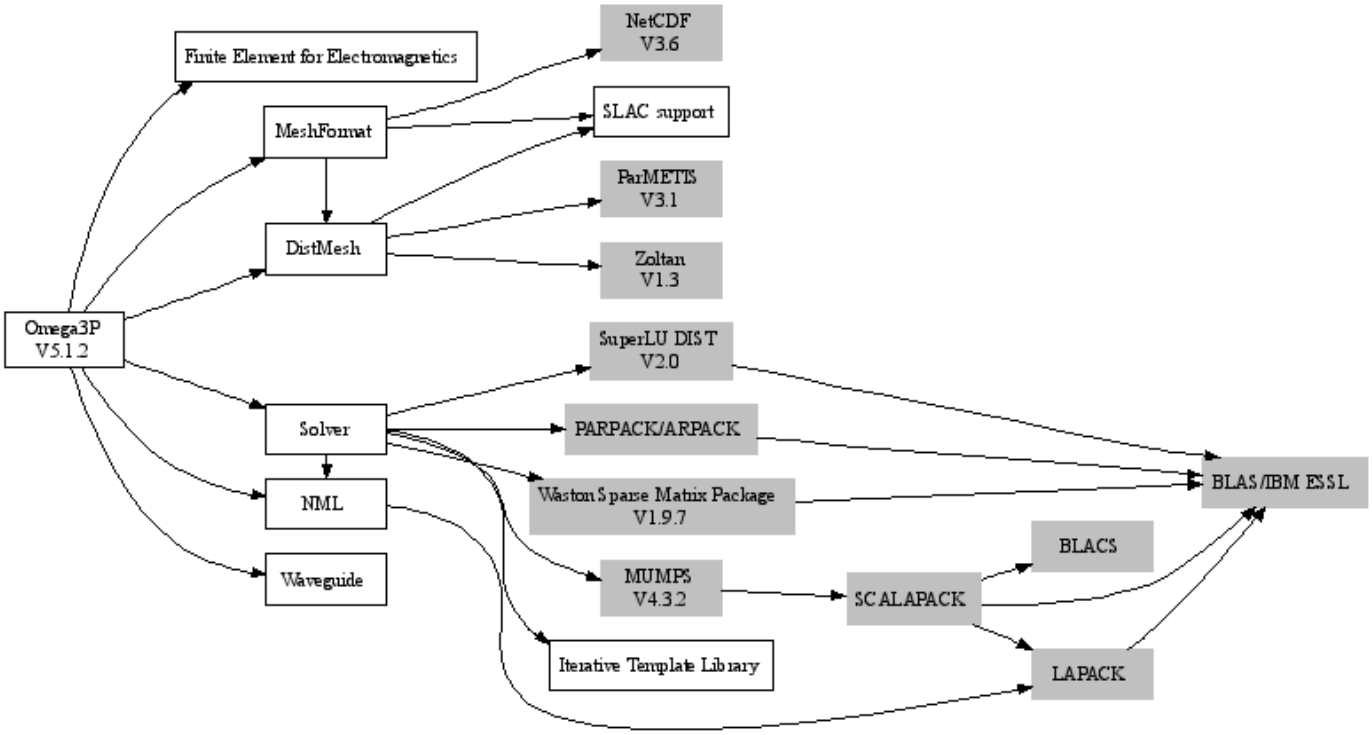
- Integrated Performance Monitoring (IPM): David Skinner, NERSC
- Tuning and Analysis Utilities (TAU): Univ. of Oregon, LANL, Research Centre Julich

Omega3P

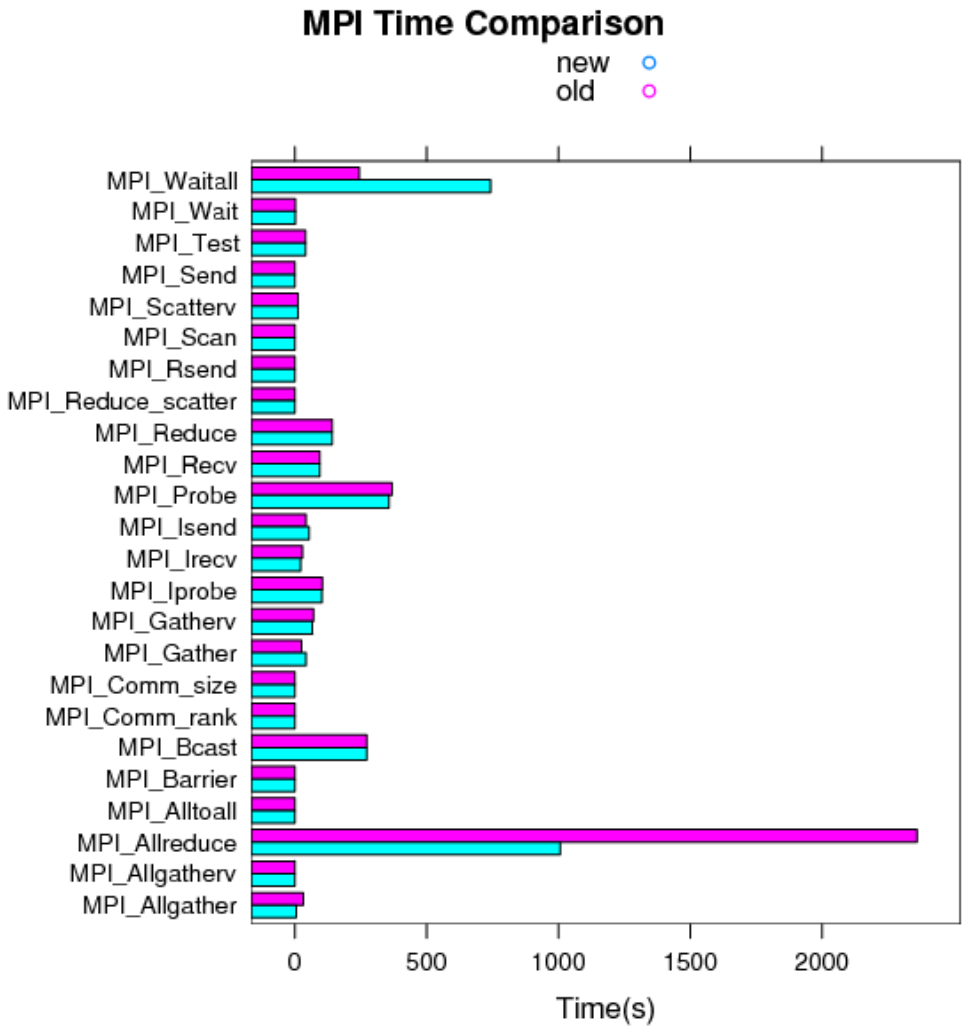
- Software developed at Stanford/SLAC (Kwok Ko and others)
- Models the 3D complex cavity of a linear accelerator
- Started with DOE Accelerator Grand Challenge, continued under SciDAC AST
- Motivation: much cheaper than alternative of manufacturing alternative designs and then discovering details of their characteristics experimentally.
- Computationally, a large high-performance high-accuracy iterative eigensolver.

Omega3P code

- Roughly 100,000 lines of C++ code
- Uses MPI for communication
- Uses SuperLU and many other libraries



Omega3P results



- On Seaborg, IPM showed that most of the time was in `MPI_Allreduce`
- Rich Lee was able to modify the algorithm to see a large speedup

BeamBeam3D

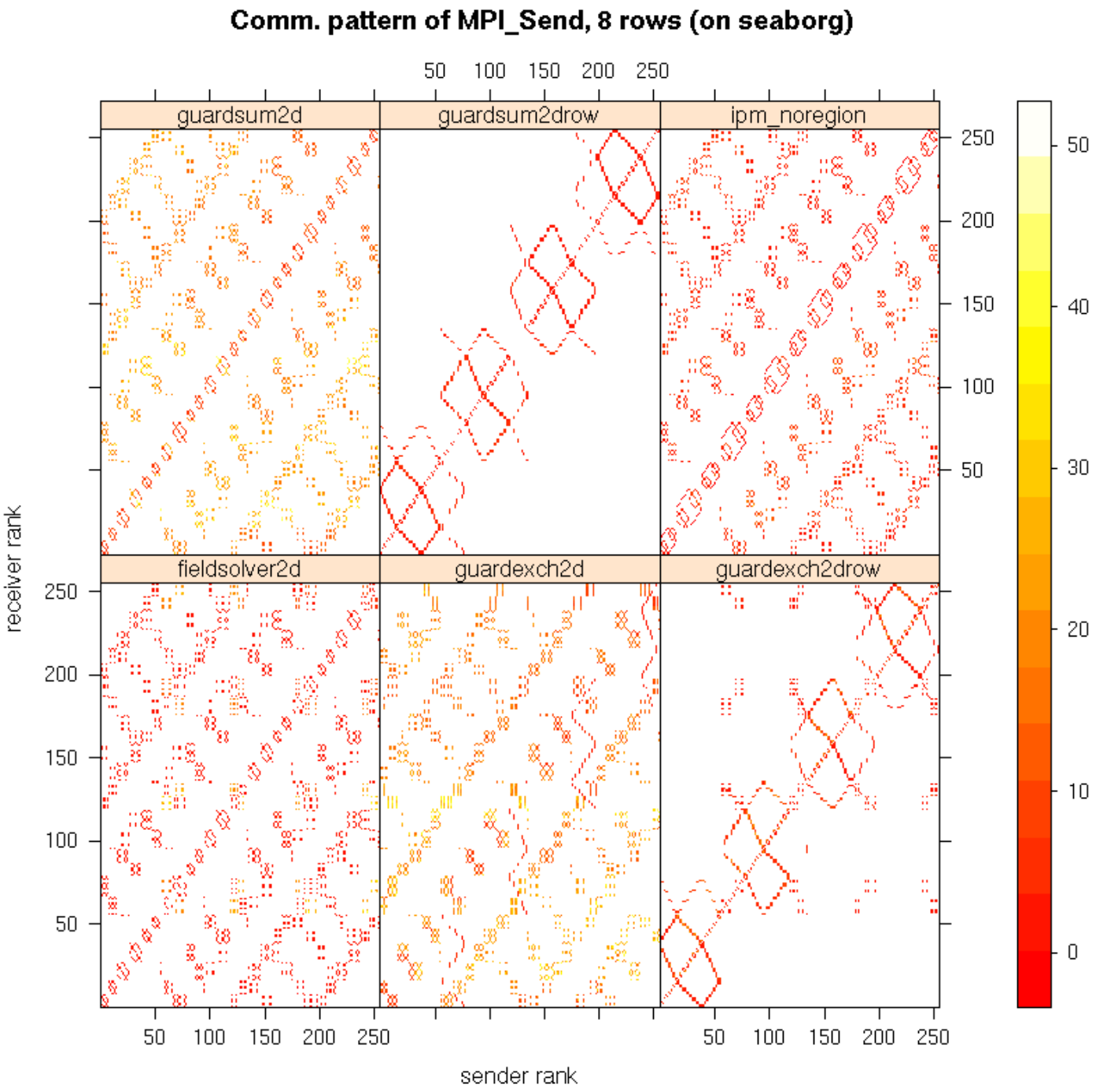
- Models collisions between two particle "beams" (in a linac)
- Developed by Rob Ryne's group at LBNL
- The code has been used to study beam-beam effects in the RHIC, Tevatron, and LHC
- Computationally, a "particle-in-cell" code that computes Poisson's equations using a parallel 2-D FFT

BeamBeam3D code

- Roughly 23,000 lines of Fortran90
- Uses MPI for communication
- Roll-your-own [Numerical Recipes] FFT (local algorithm and communication)!

BeamBeam3D results (1)

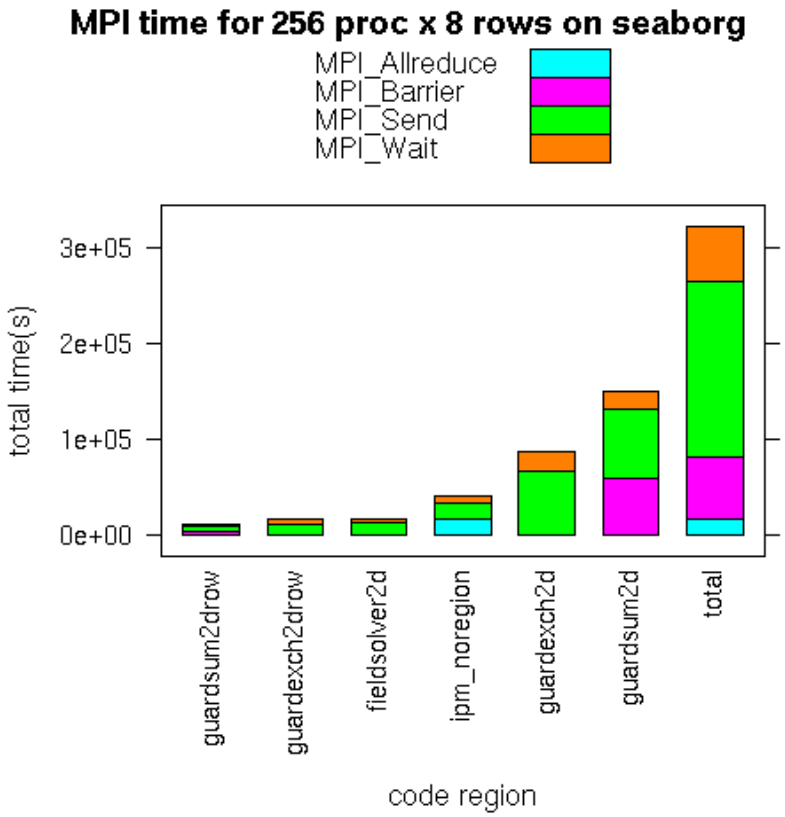
IPM + my own 'R' scripts provide an interesting picture of communication patterns



BeamBeam3D results (2)

But the most interesting result is that on 256 processors, 60-80% of the time is spent in communication! As opposed to 10-15% of the time on 32 processors.

Most of that time is spent in `MPI_SEND` and `MPI_BARRIER`



Tidbit: Looking forward -- FFT"B"?

- As BB3D suggests, there may be a need for a self-optimizing parallel FFT library.
- The state-of-the-art is FFTW, developed at MIT. But they are eliminating support for their parallel (MPI) codebase in version 3.x and future versions.
- A relatively large percentage (~50%) of important scientific codes use FFT's, and in some cases they are a significant part of the performance
- Would accrue the advantage of libraries: optimize it once, and very well.
- Faces the danger of libraries: too many variations in application requirements.

Questions?

- This provided, I hope, the flavor of PERC's activities
- Despite some frustrations, I've learned a lot about compiling, running, and analyzing HPC codes
- I'm not sure whether there are any direct tie-ins with DSD work, but at least at the software engineering level, we could probably teach eachother some things.